tokenizer.c
Naeem Hossain
September 29, 2016
Computer Architecture

A tokenizer should accept a string as a command-line argument. The string will contain one or more tokens, where each token is a either a floating-point constant, or an integer constant in hex, decimal or octal. The program generates the output “bad token” if it doesn’t recognize one of the six defined token types. The given struct TokenizerT holds two pointers, one to hold the string position at the beginning and one that parses the string. The parser pointer identifies all the tokens.

The TKGetNextToken function first sorts the pointer object as a delimiter, operator, or alphanumeric (word, decimal, floating point, hex, or octal). First, it uses an ‘if’ statement to check if the char is a word/not a word, then, it calls upon a series of helper methods to further identify.

However, if the token is an operator char, the getOpToken finds the full length of the token and retrieves its type in O(1) - O(n) runtime. To do this, getOpToken uses the sum of the ascii values as the hash key. The hash table is initialized with the type values in the InitializeTable function, since the operator is the beginning of the string value. It is programmed so that the name is parsed out of the string and returned. Using the hashtable ensures that the program will run in 0(1) time almost always. Collisions and subsequent rehashes are dealt with by changing sums so that operators have different keys.